PENERAPAN ALGORITMA QUANTUM TREEMAPS DAN BUBBLEMAPS PADA ZOOMABLE IMAGE BROWSER

Lisana
Jurusan Teknik Informatika, Fakultas Teknik, Universitas Surabaya
e-mail: lisana@ubaya.ac.id

ABSTRAK

Pada umumnya, aplikasi penampil gambar (image browser) menampilkan gambar dengan menggunakan miniatur gambar (thumbnail) yang diatur dalam bentuk grid, dimana pengguna dapat memilih gambar yang hendak dilihat dengan menggunakan mouse. Meskipun aplikasi penampil gambar sangat populer, mereka memiliki beberapa kekurangan. Misalnya, mereka hanya bisa menampilkan gambar yang terletak dalam satu direktori saja. Jika dalam direktori tersebut ada sangat banyak gambar dan tidak cukup untuk ditampilkan semuanya dalam satu layar, maka pengguna harus menggunakan scrollbar untuk melihat sisanya. Untuk beberapa kasus, miniatur gambar yang ditampilkan terlalu kecil sehingga pengguna tidak tahu apa isi dari gambar tersebut. Untuk mengatasi kekurangan-kekurangan diatas, digunakan Zoomable User Interface (ZUI) yang terintegrasi didalam sebuah aplikasi penampil gambar yang diberi nama Zoomable Image Browser yang dapat menampilkan gambar dengan menggunakan miniatur gambar tanpa scrollbar meskipun jumlahnya banyak. Selain itu, pada aplikasi ini pengguna dapat menampilkan gambar-gambar dari banyak direktori sekaligus. Untuk menyusun kelompok-kelompok gambar tersebut secara otomatis, digunakan dua algoritma, yaitu algoritma quantum treemaps dan algoritma bubblemaps. Quantum treemaps adalah algoritma yang membagi daerah yang ada dalam bentuk persegi panjang. Persegi panjang yang dihasilkan akan dibagi-bagi lagi menjadi persegi panjang yang lebih kecil dan begitu seterusnya. Algoritma bubblemaps bekerja dengan mengisi sel-sel yang ada dalam sebuah grid, menyimpan tanda dari sel-sel yang ada, sehingga dapat diketahui bahwa sel tersebut akan ditempati oleh gambar dari kelompok yang mana. Kesimpulan yang dapat diambil adalah karena algoritma bubblemaps menghasilkan daerah dengan bentuk acak, maka pengguna akan kesulitan dalam menemukan kelompok gambar tertentu, tetapi tidak menghasilkan tempat kosong pada kelompok gambar. Sedangkan algoritma quantum treemaps menghasilkan daerah berbentuk persegi panjang untuk menampung kelompok-kelompok gambar, sehingga pada sebagian besar daerah kelompok gambar terdapat tempat kosong, tetapi pengguna akan lebih mudah membedakan kelompok-kelompok gambar.

Kata Kunci*: Quantum Treemaps, Bubblemaps, Zoomable Image Browser*

1. PENDAHULUAN

Image browsing merupakan hal yang penting untuk sejumlah alasan. Alasan pertama adalah tidak peduli sistem informasi atau software apa yang digunakan, pengguna harus melihat-lihat hasil pencarian yang dilakukannya terhadap barang yang disukainya. Oleh karena itu, sangatlah penting untuk mengembangkan sistem yang dapat membantu para pengguna untuk mendapatkan hasil yang sesuai dengan keinginan pengguna. Tetapi dibutuhkan image untuk di-browse, sehingga pengguna dapat melihat-lihat image mana yang sesuai dengan keinginannya dan sistem ini dapat memberikan hasil yang terbaik.

Alasan kedua kenapa membutuhkan image browser yang baru adalah supaya lebih cerdik. Kadang-kadang orang melihat-lihat image hanya untuk kesenangan saja dan orang tersebut sering melakukannya bersama dengan yang lain. Hal ini merupakan kenyataan, terutama bagi orang yang berkecimpung dalam dunia foto. Dengan semakin banyaknya gambar yang ada, maka dibutuhkan suatu tool yang lebih baik untuk membantu pengguna dalam melihat gambar-gambar tersebut, yaitu pengguna dapat melihat gambar-gambar tersebut secara bersamaan pada sebuah layar komputer. Bagaimanapun, orang lebih suka menemukan foto yang disukainya tanpa harus mencari dari sejumlah foto yang ada.

Terdapat beberapa teknik yang dapat digunakan dalam menampilkan kumpulan gambar atau image. Oleh karena itu dapat dibuat suatu mekanisme yang dapat menyusun beberapa kumpulan gambar atau image secara otomatis dan juga memberikan interface atau tampilan yang sederhana. Adapun algoritma yang digunakan dalam menyusun gambar-gambar tersebut, yaitu algoritma quantum treemaps dan bubblemaps.

Algoritma quantum treemaps membagi daerah yang ada menjadi beberapa bagian dalam bentuk *rectangle*. Sedangkan algoritma bubblemaps membagi daerah yang ada dalam bentuk rectangular atau circular. Algoritma bubblemaps lebih efisien dalam menggunakan daerah yang ada daripada algoritma quantum treemaps, karena apabila menggunakan algoritma bubblemaps, maka tidak ada tempat kosong yang tidak terpakai pada masing-masing kelompok gambar. Lain halnya dengan algoritma quantum treemaps. Dengan menggunakan algoritma ini dapat meninggalkan tempat kosong yang tidak dipakai.

2. DASAR TEORI

2.1 *Image Browser*

Browsing secara dua dimensi (horisontal dan vertikal) dibutuhkan dalam program *painting* atau *drawing* ketika ukuran gambar atau *image* lebih besar daripada layar. Banyak desainer menggunakan *scroll bar* dua dimensi, sehingga dapat mengontrol ke arah horisontal dan vertikal. Hal ini sangat efektif jika pengguna sering menggerakkan *scroll bar* sedikit demi sedikit pada satu arah (kurang dari satu layar).

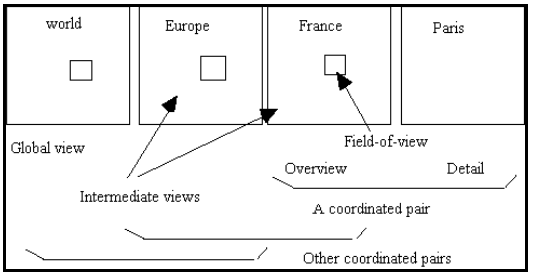
Solusi ini menjadi tidak terlalu berguna jika ukuran gambar jauh lebih besar daripada layar. Untuk menampilkan kembali gambar tersebut memerlukan waktu yang lama dan *overview, zooming, diagonal panning*, serta *multiple detail view* dibutuhkan. Kebutuhan-kebutuhan ini diperlukan oleh program drawing dan aplikasi yang lainnya.

Dalam GIS (*Geographic Information System*), pengguna dapat menjelajahi peta dunia dan mencari detail view dari sebuah negara atau kota. *Overview* yang disediakan oleh peta dunia adalah fasilitas yang sangat berguna dan diperlukan untuk membantu pengguna dalam melihat secara terperinci. Perbandingan dari *overview* ke *detail view, zooming factor* dapat bernilai 1:10, 1:10000, atau bahkan 1:1000000. Tugas GIS juga meliputi mengikuti sebuah sungai, batas negara atau kota, jalan raya (*diagonal panning*), perbandingan dari dua pelabuhan (*multiple detail view*), atau menampilkan jalan raya dan peta kepadatan populasi secara bersamaan (dua peta yang berhubungan).

Ada beberapa istilah penting dalam image browser, antara lain *detail view*, *global view*, coordinated pair of view, *intermediate view*, dan sebagainya. Berikut akan dijelaskan lebih lanjut mengenai istilah-istilah tersebut.

- *Detail view*
  Detail view (disebut juga *local view*) hanya menunjukkan sebagian dari gambar atau image, biasanya ukuran dari gambar tersebut diperbesar. Dalam aplikasi peta dunia, peta negara Paris dapat ditunjukkan dalam bentuk *detail view*. Seberapa *detail* gambar yang dibutuhkan tergantung pada tugas yang akan dilakukan.
- *Coordinated pair of view*
  Pair of view (*detail* dan *overview*) menampilkan gambar dalam *detail view* dan juga menampilkan gambar tersebut dalam *overview*. Hierarki dari tampilan biasanya disediakan oleh beberapa *coordinated pair*, dimana *detail view* dari yang satu menjadi *overview* dari yang lainnya.
- *Zooming factor (ZF)*
  Tingkat pembesaran antara dua tampilan (ZF >= 0)
- *Global view*
  Global view dapat menampilkan keseluruhan bidang yang dapat dijelajahi. Salah satu tujuan dari *global view* adalah untuk memberikan informasi apa yang ada di dalam gambar dan informasi apa yang tidak ada. Sebagai contoh, *global view* pada peta dunia (atlas) dapat menunjukkan peta dari bumi, memberikan informasi bahwa dunia ini tidak hanya terdiri dari bulan atau galaksi. Untuk mendapatkan sebanyak mungkin informasi yang ada dalam gambar tersebut, pengguna perlu untuk melihat *global view* and maksimum dari *detail view*. Maksimum dari *zooming factor* berhubungan dengan kepadatan dari gambar.
- *Intermediate view*
  Intermediate view mnunjukkan gambar antara *global view* dan *detail view* yang paling kecil.
- *Field-of-view*
  Dalam *coordinated pair*, *field-of-view* menunjukkan *overview* lokasi dan bentuk dari coordinated *detail view*.

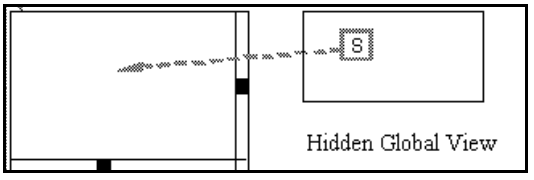
Pada Gambar 1 *Global*, *Intermediate*, dan *Detail view* ditunjukkan jika *overview* menampilkan peta Perancis, maka *detail view* akan menampilkan daerah Paris (sesuai dengan yang ditunjuk oleh *field-of-view* pada *overview*). Bagaimanapun, jika *overview* adalah seluruh dunia, maka *intermediate view* adalah Eropa dan Perancis.



Gambar 1. Global, Intermediate, dan Detail view

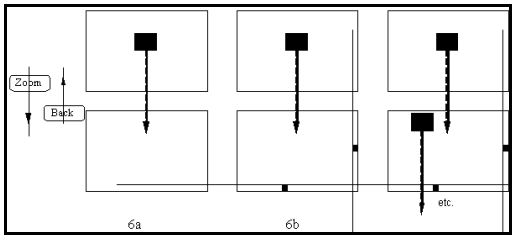
Untuk membuat *image browser* terdapat beberapa teknik dan variasi, diantaranya detail only browser, zoom and replace browser, fisheye browser, bifocal browser, dan sebagainya. Berikut akan dijelaskan lebih lanjut mengenai variasi tersebut.

- Detail only browser
  Terdapat sebuah *window* yang menunjukkan *detail view* dari gambar, dimana dapat di-*pan* baik secara horisontal maupun vertikal pada *detail view* dari gambar tersebut (Gambar 2).



Gambar 2. Single View Browser

- Zoom and replace browser
  Global view ditampilkan, maka keseluruhan gambar dapat dilihat (Gambar 3).

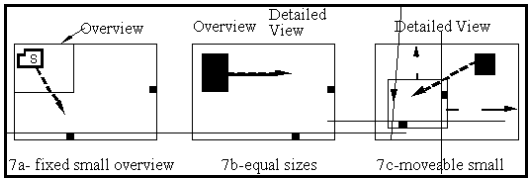


Gambar 3. Zoom dan Replace Browser

Pengguna kemudian menandai sebuah daerah dengan bentuk rectangular dan daerah tersebut diperbesar dan menggantikan gambar yang aslinya.

- Single coordinated pair

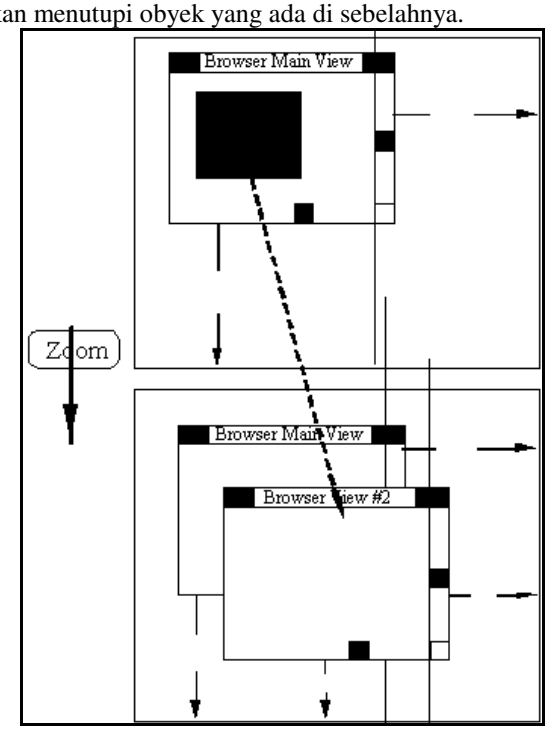
. Terdapat tiga cara pengaturan layar untuk menampilkan *overview* dan *detail view*. Cara pertama (7a) yaitu sebagian kecil dari layar digunakan untuk menampilkan *global view*, sedangkan cara kedua (7b) yaitu kedua *window* dengan ukuran yang sama digunakan untuk menampilkan *detail view* dan *overview*. Cara yang terakhir (7c) adalah *overview* dapat menempati sebagian besar dari layar, sementara *window* dengan ukuran yang kecil digunakan untuk menampilkan *detail*. Tampilan dapat dilihat pada Gambar 4.



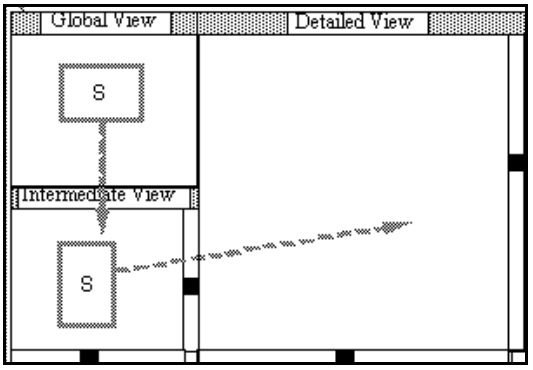
Gambar 4. Tiga Contoh Single Coordinated Pair

- Free zoom dan multiple overlap browser
  Browser Ini sesuai untuk aplikasi yang bekerja pada platform yang cepat dan layar yang besar. Pengguna bebas untuk mespesifikasikan, bergerak, membentuk kembali dan menghapus setiap *window* yang diinginkannya. Tampilan dapat dilihat pada Gambar 5.

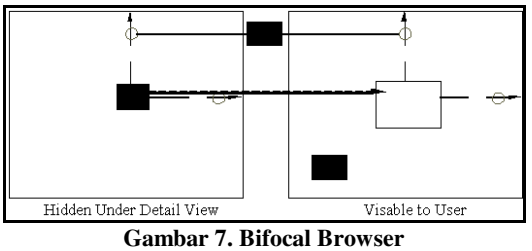
- Tiled Multilevel Browser
  Field-of-view untuk *global view* menentukan *intermediate view*. Jika pengguna mengubah *field-of-view* pada tampilan *global view* atau melakukan *scroll* pada *intermediate view*, maka tampilan dari *intermediate view* dan field-of view dari *global view* akan berubah lokasinya. Begitu juga dengan *intermediate view* dan *detail view* (Gambar 6).
- Bifocal view browser
  Menggunakan pembesaran dengan glass metaphor. Gambar yang sudah diperbesar tersebut ditaruh di atas, dimana obyek yang diperbesar tersebut berada. Oleh karena itu, gambar yang diperbesar tersebut akan menutupi obyek yang ada di sebelahnya.



Gambar 5. Overlapped Window Browser



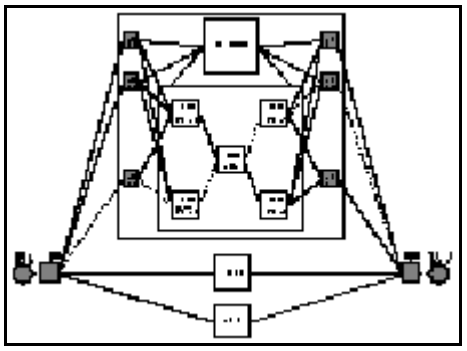
Gambar 6. Tiga Lever Browser



Gambar 7. Bifocal Browser

- Fisheye view
  Perkembangan dari bifocal. Gambar berubah bentuk, sehingga yang menarik adalah gambar ditampilkan dengan tingkat perbesaran yang tinggi, sementara gambar tersebut dipadatkan pada bagian yang lain secara bertahap. Teknik ini menggunakan satu tampilan untuk menunjukkan *global view* yang berubah bentuk, sehingga *zooming* atau *scrolling* tidak dibutuhkan. Gambar dapat berubah bentuk

secara drastis, terutama pada gambar yang berukuran besar.



Gambar 8. Fisheye View untuk Jaringan (Network)

Terdapat lima tugas utama yang diselesaikan dengan image browser. Berikut akan dijelaskan lebih lanjut mengenai tugas-tugas tersebut.

- Image generation (drawing, painting, scanning, dan sebagainya)
  Pengguna menggambar atau melukis gambar (*image*) atau diagram dalam ukuran yang besar. Perhatian pengguna berfokus pada sebagian kecil dari gambar tersebut, tetapi terkadang juga dibutuhkan untuk melihat gambar tersebut secara keseluruhan. Dengan menggunakan program menggambar, pengguna dapat berkonsentrasi pada bagian tertentu dari gambar dan kemudian dapat melihat gambar tersebut pada keseluruhan layar. Dengan program CAD/CAM, seorang desainer kapal dapat menghabiskan waktu selama satu jam untuk menggambar haluan dari kapal kemudian memeriksa bentuk keseluruhan dari lambung kapal. Dalam hal ini yang penting adalah unit dan ukuran. Untuk mengenerate gambar, *overview* sangatlah penting, tetapi kebanyakan waktu dihabiskan pada tingkat *detail*.
- Open-ended exploration
  Seorang pemain game *adventure* atau petualangan dapat bergerak dengan cepat menjelajahi wilayah-wilayah yang ada pada game tersebut, supaya pemain tersebut semakin mengenal game tersebut. Wilayah yang akan dijelajahi itu tidak diketahui sebelumnya, sehingga pemain dapat dengan mudah tersesat. *Overview* dari wilayah yang dijelajahi tersebut tidak selalu lengkap ataupun tersedia karena wilayah ini dijelajahi untuk pertama kalinya. Navigasi perlu cepat dan *interface* perlu dikuasai dengan cepat.
- Diagnostic
  Kasus khusus dalam *exploration* atau penjelajahan adalah *diagnostic*. Sebagai contoh, dengan menggunakan *pathology workstation*, seorang ahli patologi dapat memeriksa contoh digital dari jaringan pada resolusi rendah dan tinggi. Pengamatan otomatis yang lengkap dapat mengatasi kesalahan. Lokasi-lokasi yang penting dapat disimpan untuk *review* yang akan datang.
- Navigation
  Pengguna menjelajahi lingkungan yang dikenal dengan baik maupun tidak. Pertanyaannya bagaimana bisa sampai kesana? GIS (Geographical Information System) menyediakan semacam kendaraan yang dapat digunakan untuk mengantarkan truk pada tujuannya ataupun mengantarkan turis ke museum. *Global view* diperlukan untuk menunjukkan posisi yang sekarang, menyediakan hubungan atau jalur, petunjuk, dan batas pada daerah yang akan dituju. Kemudian informasi yang berhubungan ditunjukkan pada tampilan dengan tingkat pembesaran minimum yang cukup untuk menunjukkan rute ke tujuan.
- Monitoring
  Dalam hal ini, pengguna perlu untuk mengamati semuanya dan harus mengetahui status informasi dari keseluruhan sistem yang sedang diamati. Ketika terjadi masalah, pengguna harus dapat memperhatikan aspek-aspek lokal sambil melihat *overview*. *Multiple view* dapat menjadi masalah yang harus disimpan secara *global* karena jumlah dari *window* dapat menjadi sangat besar. Manajemen pada *window* adalah masalah yang penting.

2.2 *Algoritma Quantum Treemaps dan Bubblemaps*

Kolaborasi kedua algoritma tersebut digunakan dalam penyusunan kelompok gambar agar dalam satu layar dapat ditampilkan gambar sebanyak mungkin dan memanfaatkan tempat yang tersedia dengan semaksimal mungkin

Quantum treemaps adalah variasi dari algoritma treemaps yang ada. Quantum treemaps mempunyai input berupa jumlah gambar dari setiap kelompok gambar (n angka) dan sebuah *rectangle* (*Box*). Algoritma ini membagi area menjadi n *rectangle*. *Rectangle-rectangle* tersebut akan menempati *Box* yang diinputkan tadi, dimana *rectangle-rectangle* yang dihasilkan tersebut proporsional dengan jumlah gambar dari setiap kelompok gambar yang diinputkan. Berikut adalah algoritma quantum treemaps:

Input: $L_1...L_n$ serangkaian angka yang berisi jumlah gambar dari setiap kelompok gambar yang akan ditampilkan, dimana angka ini digunakan untuk menentukan ukuran dari *rectangle* yang dihasilkan.

Box suatu kotak yang digunakan untuk menyusun atau mengatur *rectangle-rectangle* ke dalamnya.

Aspect Ratio

Output: $R_1...R_n$ serangkaian *rectangle* yang memenuhi *Box* dan dimana area dari R_1 proporsional atau sebanding dengan L_1.

Algoritma ini memilih sebuah pivot, L_p dan menempatkannya ke dalam sebuah *Box*. Kemudian menyusun $L_1...L_{p-1}$ pada salah satu sisi pivot secara *recursive* dan $L_{p+1}...L_n$ pada sisi pivot yang lainnya.

1. Jika n = 1, maka hitung sebuah *rectangle* R yang mengandung L *quantum* pada susunan *grid*, dimana *grid* tersebut mempunyai sebuah *aspect ratio* yang memungkinkan untuk *Box* tersebut dan berhenti.
2. Memilih sebuah pivot, R_p. Strategi dalam memilih pivot dapat dilakukan dengan memilih tengah, elemen dengan jumlah gambar yang paling banyak, elemen yang membagi elemen dengan jumlah gambar yang sama atau hampir sama, dimana elemen ini adalah kelompok gambar yang hendak ditampilkan. R_p adalah rectangle untuk menampung kelompok gambar yang menjadi pivot.
3. Menghitung R_1, sehingga tingginya memenuhi *Box* dan lebarnya cukup untuk menampung $L_A = L_1...L_{p-1}$.
4. Bagi $L_{p+1}...L_n$ menjadi dua bagian, L_B dan L_C yang akan disusun atau diatur dalam R_2 dan R_3. Menghitung dimana *splitting point* berada, sehingga R_p mempunyai *aspect ratio* yang mendekati satu.
5. Menghitung R_p, R_2 dan R_3. Hal ini dilakukan dengan menggunakan *ratio* diantara ukuran dari kelompok gambar yang bersangkutan dan memecah tempat kosong yang ada dengan menggunakan *ratio* yang sama.
6. Lakukan langkah 2 sampai langkah 5 secara *recursive* untuk L_A pada R_1, L_B pada R_2, dan L_C pada R_3.
   a. Membentuk *rectangle-rectangle* dalam R_p, R_2 dan R_3 untuk menghindari overlapping R_1 atau satu dengan yang lainnya.
   b. Memastikan bahwa R_p dan R_2 mempunyai tinggi yang sama. Memastikan bahwa R_p dan R_2 mempunyai tinggi yang sama dengan R_1. Langkah-langkah algoritma ini juga dapat berhenti apabila ditemukan suatu daerah atau bagian yamg sudah terlalu kecil. Apabila lebarnya tidak mencukupi, maka tambahkan jumlah tambahan pada lebar dari *rectangle* yang ada dalam daerah tersebut yang menyentuh atau menekan batas kanan dari daerah tersebut. Lakukan hal yang sama pada *rectangle* yang tingginya tidak mencukupi.

Algoritma quantum treemaps diatas digunakan untuk bentuk regular. Pada perkembangan algoritma quantum treemaps berikutnya dihasilkan bentuk baru bagi algoritma ini, yaitu quad dan snake. Dimana kedua bentuk tersebut digunakan untuk sejumlah kecil gambar. Maka algoritma baru adalah sebagai berikut:

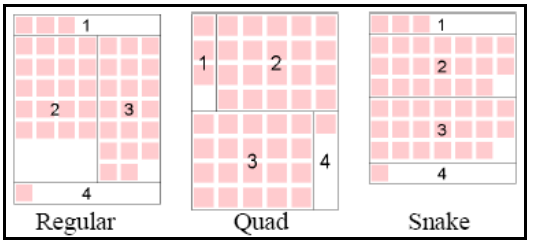
new 1a. Jika n = 4, maka pertama-tama mencoba pengaturan seperti biasa dengan melanjutkan rekursi ke langkah selanjutnya.

new 1b. Jika n = 4, maka keempat kelompok gambar tersebut akan diatur secara *quad*. Membagi *Box* menjadi dua pada horisontal atau vertikal (tergantung orientasi dari *Box*) berdasarkan jumlah elemen yang terdapat di dalam empat kelompok gambar tersebut. Kemudian, setiap *rectangle* tersebut dibagi lagi menjadi dua pada orientasi yang berlawanan dari orientasi sebelumnya berdasarkan jumlah elemen dari dua kelompok gambar tersebut. Maksudnya jika sebelumnya *rectangle* dibagi secara horisontal, maka berikutnya *rectangle* dibagi secara vertikal.

new 1c. Jika n = 4, maka keempat kelompok gambar tersebut akan diatur secara *snake* dengan membagi *Box* menjadi 4 kotak (secara horisontal atau vertikal, tergantung dari orientasi dari Box) berdasarkan jumlah elemen yang ada di dalam 4 kelompok gambar tersebut.

new 1d. Menghitung *aspect ratio* dan tempat kosong yang terbuang dari keempat *rectangle* yang telah dihasilkan tersebut dari langkah 1a, 1b dan 1c, dan gunakan pengaturan yang terbaik dari seluruh hasil tersebut.

Hasil dari algoritma quantum treemaps dengan menggunakan algoritma yang baru dapat dilihat pada Gambar 9.



Gambar 9. Algoritma Quantum Treemaps

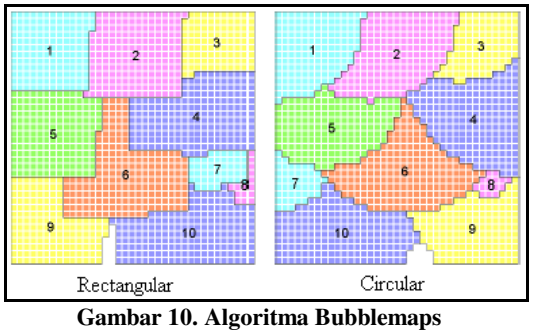
Algoritma bubblemaps ini bekerja dengan mengisi *cell-cell* yang ada dalam sebuah *grid*, menyimpan tanda dari *cell-cell* yang ada, sehingga dapat diketahui bahwa *cell* tersebut akan ditempati oleh gambar dari kelompok yang mana. Algoritma ini mengisi *cell-cell* satu kelompok gambar dalam suatu waktu. Dengan menggunakan algoritma yang berbeda untuk memilih *cell* berikutnya yang akan ditempati, bentuk dari kelompok-kelompok gambar tersebut dapat diatur, yaitu rectangular (seperti kotak) atau circular (seperti lingkaran). Berikut adalah algoritma dari bubblemaps:

Input: $L_{1...}L_n$, Aspect Ratio

1. Hitung ukuran dari seluruh *grid* berdasarkan jumlah total dari gambar atau *image* yang akan disusun atau diatur dan hasil aspect ratio yang diinginkan.

2. Membuat sebuah grid yang sesuai dengan ukuran yang telah dihitung pada langkah pertama dan setiap *cell* diberi nilai UNASSIGNED.
3. Untuk setiap kelompok atau kelompok dari gambar tersebut, L_i, panggil algoritma yang ada, mulai dari langkah keempat dan kemudian berhenti.
4. Menemukan *starting point* atau point awal yang diinginkan dengan mencari *cell* UNASSIGNED pertama dalam *grid* (dengan menggunakan urutan kiri-kanan, atas-bawah). Kemudian inisialisasi sebuah daftar dari *cell-cell*, yang disebut dengan LIST dan tambahkan *starting point* atau point awal tersebut ke dalam LIST.
5. Jika LIST kosong, maka berhenti. Tetapi apabila LIST tidak kosong, maka ambil elemen pertama, P, hapus P dari LIST, dan *cell* yang ada pada lokasi P diberi nilai ASSIGNED.
6. Untuk setiap *cell* UNASSIGNED yang merupakan tetangga dari P, Q, masukkan Q ke dalam LIST dan *cell* yang ada pada lokasi Q diberi nilai dengan ID dari kelompok yang sekarang sedang dikerjakan. Kemudian langkah berikutnya kembali ke langkah lima.

Catatan bahwa urutan dalam memasukkan tetangga dari P yang disebut Q ke dalam LIST mempengaruhi bentuk yang dihasilkan untuk kelompok gambar tersebut. Artinya dengan menggunakan algoritma yang berbeda dalam memilih *cell* berikutnya yang akan dimasukkan ke dalam LIST, maka bentuk dari kelompok gambar tersebut dapat diatur. Oleh karena itu, ada beberapa bentuk yang dapat dihasilkan apabila menggunakan algoritma ini, yaitu *rectangular* dan *circular*. Hasil dari algoritma bubblemaps dapat dilihat pada Gambar 10.



Gambar 10. Algoritma Bubblemaps

3. ANALISA DAN DESAIN

Aplikasi image browser pada dasarnya berfungsi sebagai alat untuk mengeksplorasi file-file gambar atau *image* yang terdapat pada media penyimpanan yang ada. Adapun fungsi-fungsi dasar yang harus ada pada suatu image browser pada umumnya adalah sebagai berikut:

- Dapat menampilkan semua file-file gambar atau *image* pada suatu direktori yang aktif dan menyusun atau mengatur gambar-gambar tersebut berdasarkan susunan tertentu di suatu *viewport*.
- Dapat menampilkan seluruh struktur *tree directory* dari media penyimpanan yang ada.
- Dapat membentuk visualisasi *slide show* untuk melihat seurutan file-file gambar dengan delay tertentu.
- Dapat menampilkan sebuah file gambar secara utuh (*full screen*) yang dipilih oleh pengguna dari sekumpulan file-file gambar pada *viewport*.
- Dapat menampilkan informasi metadata dari file-file gambar tersebut seperti dimensi dari gambar, tanggal pembuatan, dan sebagainya.
- Dapat mengubah orientasi dari masing-masing file gambar seperti rotasi berdasarkan sudut tertentu maupun flip secara horisontal atau vertikal.

Pembuatan Zoomable Image Browser ini menggunakan aplikasi bernama Photomesa yang dipakai sebagai acuan utama. Photomesa adalah suatu aplikasi image browser yang sudah menggunakan pendekatan zoomable image browser, dimana photomesa ini menggunakan algoritma quantum treemaps dalam mengatur atau menyusun kelompok-kelompok gambar dari direktori-direktori yang dipilih oleh pengguna.

Photomesa ini memungkinkan pengguna untuk melihat gambar-gambar dari banyak direktori sekaligus, dimana gambar-gambar tersebut dapat diperbesar, dan menggunakan mekanisme navigasi yang sederhana. Aplikasi ini juga dapat mengelompokkan gambar berdasarkan metadata yang tersedia dalam file dari sistem. Aplikasi ini hanya membutuhkan sekumpulan gambar pada suatu media penyimpanan dan tidak meminta pengguna untuk menambahkan metadata apapun, atau memanipulasi gambar-gambar sebelum dijelajahi atau di-*browsing*.

Gambar 11. Tampilan Photomesa

Dalam mengatur atau menyusun kelompok-kelompok gambar dari direktori yang dipilih, photomesa menggunakan algoritma quantum treemaps. Gambar-gambar tersebut dapat dikelompokkan menurut direktori masing-masing atau berdasarkan metadata dari file gambar tersebut.

Aplikasi zoomable image browser ini dirancang dengan menggunakan Photomesa sebagai referensi untuk menentukan fitur-fitur yang akan disediakan. Fitur-fitur yang akan dibuat antara lain:

- Tree directory
  Menampilkan seluruh struktur *tree directory* dari media penyimpanan.
- Rotate right dan rotate left
  Gambar yang dipilih dapat diputar ke kanan atau ke kiri.
- Flip horisontal dan flip vertikal
  Gambar yang dipilih akan ditampilkan dengan cara dicerminkan secara horisontal atau vertikal.
- Mengurutkan thumbnail dan kelompok gambar
  Dimana kelompok-kelompok gambar dan gambar-gambar yang ada di dalam kelompok tersebut dapat diurutkan berdasarkan tanggal (dari yang terbaru ke yang terlama atau sebaliknya) dan nama file (ascending atau descending)
- Mengelompokkan thumbnail
  Dimana gambar-gambar yang ada di dalam direktori dapat dikelompokkan berdasarkan bulan, tahun, direktori ataupun tanpa ada pengelompokan.
- Informasi metadata
  Menampilkan informasi metadata dari file-file gambar tersebut seperti nama file, lokasi gambar tersebut di harddisk, ukuran gambar, dimensi gambar, dan sebagainya.
- Menampilkan slide show
  Jika fitur ini dipilih, maka akan ditampilkan semua gambar yang ada pada layar secara otomatis bergantian dengan jangka waktu tertentu.
- Full Screen
  Menampilkan gambar yang dipilih oleh pengguna dari sekumpulan gambar pada layar secara utuh (*full screen*).
- Menampilkan foto berikutnya atau sebelumnya
  Fitur ini dapat dilakukan jika sebuah gambar ditampilkan pada satu layar penuh. Kemudian pengguna dapat memilih untuk menampilkan gambar berikutnya atau sebelumnya, sehingga gambar yang ditampilkan akan berubah.
- Memilih semua gambar atau tidak memilih semua gambar
  Fitur ini digunakan untuk membantu pengguna dalam memilih semua gambar atau tidak, sehingga pengguna dapat melakukan fitur yang lainnya dengan cepat. Misalnya, pengguna ingin melakukan rotate left atau rotate right pada semua gambar.
- Preference
  Pada fitur ini, pengguna dapat mengatur beberapa hal yang berhubungan dengan gambar-gambar yang ditampilkan tersebut. Ada tiga hal yang dapat diatur oleh pengguna, yaitu jarak antara gambar-gambar yang ditampilkan, ukuran dari kotak merah yang digunakan untuk *zoom in*, dan kecepatan *zooming* (*zoom in* dan *zoom out*).
- Search
  Pengguna dapat mencari gambar-gambar tertentu berdasarkan nama file, bulan dan tahun dari tanggal modifikasinya. Gambar-gambar akan ditampilkan berdasarkan kriteria pencarian yang dipilih oleh pengguna.
- Membuat dan menyimpan library baru dan membuka library
  Pengguna dapat memilih direktori-direktori yang berisi gambar-gambar, dimana nantinya sekumpulan direktori tersebut dapat disimpan sebagai library. Library tersebut juga dapat dibuka kembali.

4. IMPLEMENTASI

Pembuatan aplikasi zoomable image browser ini menggunakan Borland Delphi 7 beserta gabungan dari dua komponen tambahan yang bernama DrawFace dan CommonCAD. Komponen DrawFace ini *shareware*, sedangkan komponen CommonCAD bersifat *freeware*. Alasan penggabungan kedua komponen tersebut adalah karena komponen DrawFace bersifat *shareware* dan tidak menyediakan source code untuk fasilitas lainnya yang ada di DrawFace, tetapi menyediakan *source code* dasar penggambaran beberapa obyek, seperti *rectangle, ellipse, polyline, text, dan picture*. Sedangkan pada komponen CommonCAD yang bersifat *freeware*, terdapat *source code* untuk semua fasilitas yang ada pada komponen tersebut, tetapi komponen ini tidak menyediakan *source code* penggambaran obyek *picture*. Oleh karena itu, penggambaran obyek *picture* pada komponen DrawFace digabung ke dalam komponen CommonCAD.

Komponen DrawFace dipakai untuk penggambaran *image* atau *picture* di kanvas. Untuk penggambaran *picture* digunakan GDI+ dan di dalam GDI+ terdapat fasilitas untuk melakukan rotate dan flip. Selain itu, juga terdapat fasilitas untuk mendapatkan thumbnail dari gambar, sehingga untuk me-*load* gambar dan melakukan fasilitas *zooming* dapat dilakukan dengan cepat. Sedangkan fitur-fitur dari komponen CommonCAD yang digunakan dalam pembuatan aplikasi ini adalah penamaan obyek, dapat mengenali obyek yang ditunjuk oleh kursor mouse (pada status bar akan muncul nama dari obyek tersebut) dan zoom dengan menggunakan mouse wheel.

Dalam implementasi zoomable image browser dengan menggunakan algoritma quantum treemaps dan bubblemaps ini digunakan tiga form yang mewakili tiga unit. Dan terdapat empat unit lagi tanpa form. Masing-masing unit memiliki fungsinya masing-masing dalam pengimplementasian zoomable image browser. Berikut ini adalah penjelasan dari masing-masing unit:

- Unit Bubblemap
  Merupakan unit yang berisi implementasi dari algoritma bubblemaps, baik untuk mode circular

maupun rectangular. Unit ini menyimpan struktur data dan fungsi serta prosedur yang berkaitan dengan algoritma bubblemaps.

- Unit Quantum Treemap
  Merupakan unit yang berisi implementasi dari algoritma quantum treemaps, untuk mode regular, quad, dan snake. Unit ini menyimpan struktur data dan fungsi serta prosedur yang berkaitan dengan algoritma quantum treemaps.
- Unit CompPicture
  Merupakan unit yang berisi struktur data dan fungsi-fungsi serta prosedur yang berhubungan dengan komponen yang digunakan dalam aplikasi ini. Ada tiga komponen yang digunakan dalam aplikasi ini yaitu rectangle, polyline, dan picture.
- Unit CommonCAD
  Merupakan unit yang berfungsi untuk menyimpan struktur data serta fungsi-fungsi dan prosedur yang berhubungan dengan canvas dan manipulasinya. Dimana pada canvas ini, akan ditampilkan komponen-komponen yang digunakan dalam aplikasi ini, yaitu rectangle, polyline, dan picture. Unit ini akan memanggil unit CompPicture.
- Unit Main
  Merupakan unit yang dimiliki oleh Form Main. Dimana form ini merupakan form tampilan utama dalam apliksasi zoomable image browser ini. Unit ini berisi implementasi dari fitur-fitur yang tersedia dalam aplikasi ini. Segala yang berhubungan dengan fitur-fitur yang tersedia dalam aplikasi ini, dikendalikan oleh unit ini.
- Unit Full Screen
  Merupakan unit yang dimiliki oleh Form Full Screen. Dimana form ini digunakan untuk menampilkan gambar dalam versi full screen. Unit ini menyimpan fungsi-fungsi dan prosedur yang digunakan untuk segala manipulasi dari form Full Screen.
- Unit Help
  Merupakan unit yang dimiliki oleh Form Help, dimana form ini digunakan untuk memberikan bantuan dan petunjuk kepada pengguna mengenai cara pemakaian program aplikasi ini.

Berikut adalah salah satu contoh kode sumber implementasi algoritma bubblemap:

Kode sumber pembentukan grid BubbleMap

```
1:  area := computeSize(origSizes);
2:  ar := computeAspectRatio(origBox) / origiar;
3:  numRows := Ceil(sqrt(area / ar));
4:  numCols := Ceil(area / numRows);
5:  box := rect(origBox.Left, origBox.Top, (origBox.Left +
    numCols) , (origBox.Top + numRows));
6:  setLength(idGrid, numRows, numCols);
7:  for i := 0 to numRows-1 do
8:    for j := 0 to numCols-1 do
9:      idGrid[i,j] := UNASSIGNED;
10: idList  := TList.Create;
11: nextList := TList.Create;
12: for i := 0 to high(origSizes) do begin
13:   pts := fill(i, origSizes[i]);
14:   idList.add(pts);
15: end;
16: result := idList;
```

5. PENGUJIAN

Uji coba akan dilakukan pada fitur-fitur yang tersedia di dalam aplikasi ini. Beberapa fitur yang akan diuji coba adalah sebagai berikut:

- Algoritma
  Menyusun atau mengatur gambar-gambar dari direktori-direktori yang dipilih oleh pengguna.
- Sort by
  Mengurutkan kelompok-kelompok gambar dan gambar-gambar yang ada di layar berdasarkan kriteria pengurutan yang tersedia.
- Group by
  Mengelompokkan gambar-gambar yang ada di layar berdasarkan kriteria pengelompokan yang tersedia.
- Zooming
  Melakukan zoom in atau zoom out pada gambar-gambar yang dipilih.
- Find
  Melakukan pencarian terhadap file-file gambar yang tampil di layar berdasarkan inputan dari pengguna.
- Open Library dan Save Library
  Menyimpan gambar-gambar yang ditampilkan sekarang dan dapat membukanya kembali suatu saat.

Untuk setiap fitur-fitur yang akan diuji coba tersebut akan dilakukan uji coba untuk setiap variasi jumlah input yang akan digunakan. Dimana pada program zoomable image browser ini, inputnya berupa file-file gambar. Adapun variasi jumlah file gambar yang akan digunakan dalam uji coba ini adalah sebagai berikut:

- Ukuran Kecil, input berjumlah kurang dari 100 file gambar.
- Ukuran Medium, input berjumlah sekitar 100 sampai 250 file gambar.
- Ukuran Besar, input berjumlah sekitar 251 sampai 650 file gambar.
- Ukuran Sangat Besar, input berjumlah lebih dari 650 file gambar.

Performa dari aplikasi ini akan banyak bergantung pada kualitas komputer yang digunakan untuk menjalankannya. Berikut ini adalah spesifikasi dari komputer yang digunakan dalam uji coba :

- Processor AMD Turion 1.80 Ghz.
- Memory 512 MB.
- Video Card Memory 128 MB.
- Resolusi layar 1280 x 768 pixel dengan *refresh rate* monitor 60 Hertz.

Hasil uji coba menunjukkan bahwa Algoritma bubblemaps dapat memanfaatkan tempat yang ada (layar) lebih efektif jika dibandingkan dengan algoritma quantum treemaps, sehingga ukuran gambar yang dihasilkan dalam algoritma bubblemaps

terkadang lebih besar jika dibandingkan dengan algoritma quantum treemaps, walaupun jumlah gambar yang digunakan sama. Hal ini disebabkan karena pada algoritma bubblemaps daerah yang dihasilkan untuk menampung kelompok gambar berbentuk acak, yaitu rectangular dan circular, sehingga tidak ada tempat kosong pada daerah kelompok gambar, sedangkan pada algoritma quantum treemaps, daerah untuk menampung kelompok gambar pasti berbentuk rectangle, sehingga pada sebagian besar kelompok gambar tersebut pasti terdapat tempat kosong karena jumlah gambarnya lebih sedikit dari tempat yang disediakan.

Sedangkan untuk uji coba waktu yang dibutuhkan oleh algoritma bubblemaps dan quantum treemaps dalam menampilkan gambar-gambar dari direktori yang dipilih oleh pengguna tidak jauh berbeda. Hal ini dikarenakan struktur data yang digunakan oleh kedua algoritma tersebut hampir sama, walaupun cara kerja dari kedua algoritma tersebut berbeda.

6. SIMPULAN

Dengan adanya implementasi dan uji coba yang dilakukan pada pembuatan zoomable image browser dengan menggunakan algoritma quantum treemaps dan bubblemaps, dapat disimpulkan bahwa:

- Algoritma bubblemaps menghasilkan daerah dengan bentuk yang acak (*rectangular* atau *circular*) untuk menampung kelompok-kelompok gambar, sehingga pengguna akan mengalami kesulitan dalam mencari kelompok gambar tertentu secara visual. Namun dengan bentuk acak ini, algoritma bubblemaps tidak menghasilkan tempat kosong pada daerah kelompok gambar, hanya pada keseluruhan tempat yang tersedia.
- Jika menggunakan algoritma quantum treemap, pengguna akan lebih mudah dalam mencari kelompok gambar tertentu secara visual karena algoritma ini menghasilkan daerah berbentuk *rectangle* untuk menampung kelompok-kelompok gambar. Tetapi bentuk rectangle ini menyebabkan sebagian besar dari daerah kelompok gambar terdapat tempat kosong yang terbuang.
- Algoritma bubblemaps lebih tepat digunakan untuk aplikasi yang bertujuan untuk mempunyai gambar-gambar yang berhubungan berdekatan satu dengan yang lain. Sedangkan algoritma quantum treemaps lebih tepat digunakan untuk aplikasi yang dapat membedakan kelompok-kelompok gambar secara jelas.
- Aplikasi zoomable image browser yang dibuat dirancang dengan menggunakan Photomesa sebagai model. Fitur-fitur utama berhasil diimplementasikan dengan baik. Beberapa kekurangan pada aplikasi ini misalnya pada zooming (zoom in dan zoom out), bagian yang akan di-zooming harus digeser terlebih dahulu ke tengah layar, baru kemudian di-zooming. Proses pengelompokan, pengurutan, dan pencarian gambar lebih lambat jika dibandingkan dengan Photomesa karena Photomesa menggunakan database untuk menyimpan thumbnail gambar dari direktori yang sudah dipilih, sehingga untuk me-load gambar yang sama akan menjadi lebih cepat.

7. DAFTAR PUSTAKA

[1] Bederson, B. B. (2001). *PhotoMesa: A Zoomable Image Browser Using Quantum Treemaps and Bubblemaps*. UIST 2001, ACM Symposium on User Interface Software and Technology, CHI Letters, 3(2), pp. 71-80. http://www.cs.umd.edu/hcil/photomesa/2001-10.pdf

[2] Bederson, B.B., Hollan, J.D., Stewart, J., Rogers, D., Vick, D., Ring, L.T., Grose, E.,Forsythe, C. 1998. *A Zooming Web Browser*.http://www.cs.umd.edu/hcil/pad++/papers/bookchap-98-webbrowser/

[3] Bederson, B. B., Meyer, J., & Good, L. (2000). Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java. *In Proceedings of User Interface and Software Technology (UIST 2000)* ACM Press, pp. 171-180

[4] Bederson, B. B., Shneiderman, B., & Wattenberg, M. (2002) *Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies*, ACM Transactions on Graphics (TOG), 21, (4), October 2002, pp. 833-854. http://www.cs.umd.edu/hcil/photomesa/2001-18.pdf

[5] Combs, T., T.A., and Bederson, B.B. 1999. *Does Zooming Improve Image Browsing*. http://www.cs.umd.edu/hcil/piccolo/learn/papers/HCIL-99-05.pdf

[6] Kang, H., & Shneiderman, B. (2000). Visualization Methods for Personal Photo Collections Browsing and Searching in the PhotoFinder. *In Proceedings of IEEE International Conference on Multimedia and Expo (ICME2000)* New York: IEEE, pp. 1539-1542.

[7] Meyer, J., Perlin, K., Bederson, B.B., Hollan, J. 1995. *Two Document Visualization Techniques for Zoomable Interfaces*. http://www.cs.umd.edu/hcil/jazz/learn/papers/unpub-95-docvis.pdf

[8] Shneiderman, B., & Wattenberg, M. (2001). *Ordered Treemap Layouts.* Tech Report CS-TR-4237, Computer Science Dept., University of Maryland, College Park, MD.